**Bundelkhand Institute of Engineering and Technology, Jhansi ( B.I.E.T. ,JHANSI )**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**DATA STRUCTURE LAB(KCS-351) Academic Year : 2021-2022**

**SUBMITTED TO :**  Mr. SHARAD YADAV

**SUBMITTED BY:**

ANKUSH KUMAR KUSHWAHA (2004310015)

**Question 1:**  Write a program for matrix multiplication.

**CODE:**

#include<bits/stdc++.h> int N=100; using namespace std; int main()

{

int n;

cin>>n; int a[n][n]; int b[n][n]; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cin>>a[i][j];

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cin>>b[i][j];

}

}

cout<<"Matrix A:"<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<a[i][j]<<" ";

}

cout<<endl;

}

cout<<endl<<"Matrix B:"<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<b[i][j]<<" ";

}

cout<<endl;

}

int c[n][n]; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

c[i][j]=0;

for(int k=0;k<n;k++)

{

c[i][j]+=(a[i][k]\*b[k][j]);

}

}

}

cout<<endl<<"The resultant matrix C: "<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<c[i][j]<<" ";

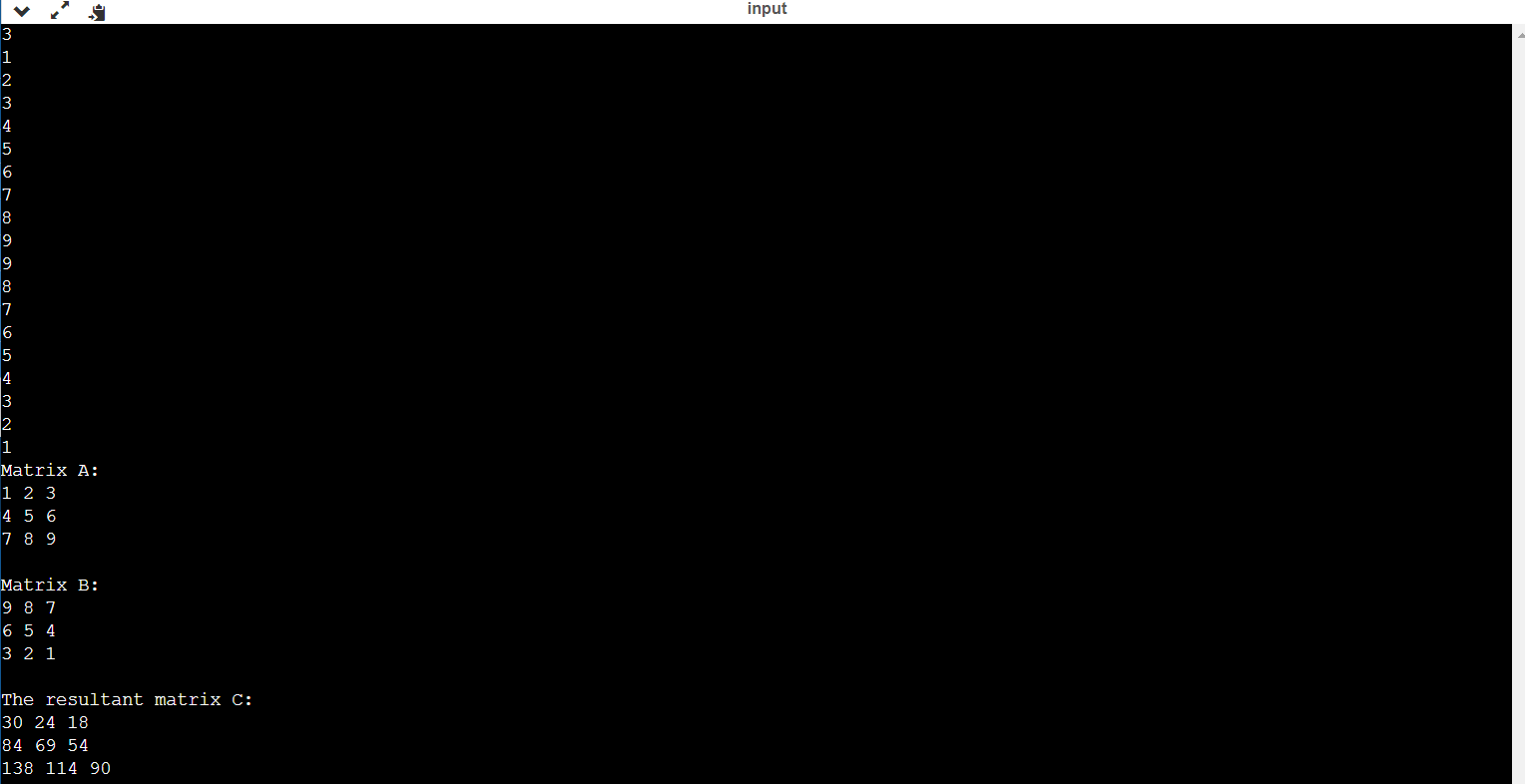
}

cout<<endl;

}

}

INPUT AND OUTPUT:



**QUESTION 2:**  Write a program to find the addition of two matrices.

CODE:

#include<bits/stdc++.h> int N=100; using namespace std; int main()

{

int n;

cin>>n; int a[n][n]; int b[n][n]; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cin>>a[i][j];

}

}

for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cin>>b[i][j];

}

}

cout<<"Matrix A:"<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<a[i][j]<<" ";

}

cout<<endl;

}

cout<<endl<<"Matrix B:"<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<b[i][j]<<" ";

}

cout<<endl;

}

int c[n][n]; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

c[i][j]=a[i][j]+b[i][j];

}

}

cout<<endl<<"The resultant matrix C: "<<endl; for(int i=0;i<n;i++)

{

for(int j=0;j<n;j++)

{

cout<<c[i][j]<<" ";

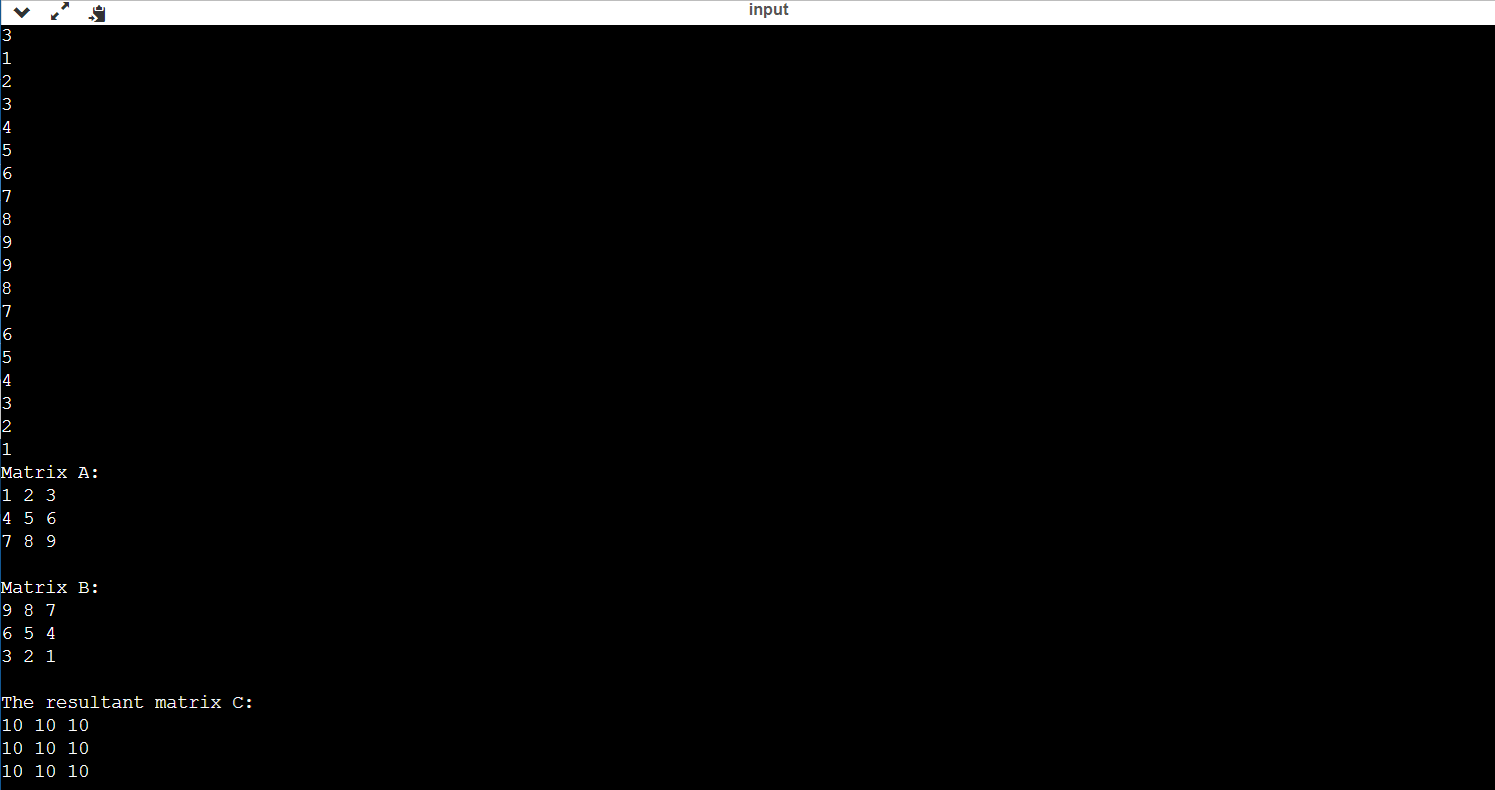
}

cout<<endl;

}

}

OUTPUT:



**QUESTION 3:**  Write a program to sort an array using Bubble Sort. CODE

#include<bits/stdc++.h> using namespace std; int main()

{

int n;

cin>>n; int a[n]; for(int i=0;i<n;i++)

{

cin>>a[i];

}

for(int i=0;i<n-1;i++)

{

for(int j=0;j<n-i-1;j++)

{

if(a[j]>a[j+1])

{

swap(a[j],a[j+1]);

}

}

}

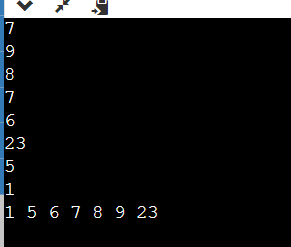
for(int i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

}

**INPUT AND OUTPUT: **

**RECURSIVE:**

#include<bits/stdc++.h> using namespace std; void bubbleSort(int arr[], int n)

{

if (n == 1)

return;

for (int i=0; i<n-1; i++) if (arr[i] > arr[i+1]) swap(arr[i], arr[i+1]);

bubbleSort(arr, n-1);

}

int main()

{

int n;

cin>>n; int a[n]; for(int i=0;i<n;i++)

{

cin>>a[i];

}

bubbleSort(a,n); for(int i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

}

INPUT AND OUTPUT:

Question 4: Write a program to sort an array using insertion sort.

CODE:

#include<bits/stdc++.h> using namespace std;

int main()

{

int n;

cin>>n; int a[n]; for(int i=0;i<n;i++)

{

cin>>a[i];

}

for(int i=1;i<n;i++)

{

int key = a[i];

int j = i - 1;

while (j >= 0 && a[j] > key)

{

a[j + 1] = a[j]; j = j - 1;

}

a[j + 1] = key;

}

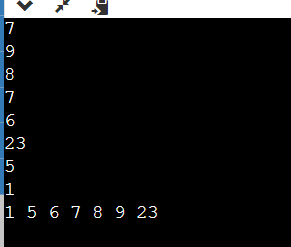
for(int i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

}

INPUT AND OUTPUT: ****

Recursive Code:

#include<bits/stdc++.h> using namespace std;

void insertionSortRecursive(int arr[], int n)

{

if (n <= 1)

return;

insertionSortRecursive( arr, n-1 ); int last = arr[n-1]; int j = n-2; while (j >= 0 && arr[j] > last)

{

arr[j+1] = arr[j];

j--;

}

arr[j+1] = last;

}

int main()

{

int n;

cin>>n; int a[n]; for(int i=0;i<n;i++)

{

cin>>a[i];

}

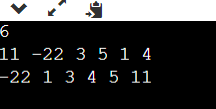
insertionSortRecursive(a,n); for(int i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

}

INPUT AND OUTPUT: 

Question 5: Write a program to sort an array using Selection Sort.

CODE:

#include<bits/stdc++.h> using namespace std;

int main()

{

int n;

cin>>n; int a[n]; for(int i=0;i<n;i++)

{

cin>>a[i];

}

int i, j, min\_idx;

for (i = 0; i < n-1; i++)

{

min\_idx = i; for (j = i+1; j < n; j++) if (a[j] < a[min\_idx])

min\_idx = j;

swap(a[min\_idx], a[i]);

}

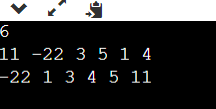
for(int i=0;i<n;i++)

{

cout<<a[i]<<" ";

}

}

INPUT AND OUTPUT: 

Question 6: Write a program to find a target element in a given array using linear Search.

CODE:

#include<bits/stdc++.h> using namespace std; int linearSearch(int arr[],int key ,int n)

{

for(int i=0;i<n;i++)

{

if(arr[i]==key)

{

return i;

}

}

return -1;

}

int main()

{

int n;

cin>>n; int a[n];

for(int i=0;i<n;i++)

{

cin>>a[i];

}

int target; cin>>target; int status=linearSearch(a,target,n); if(status!=-1)

{

cout<<"Target element is present at the position :"<<status<<endl;

}

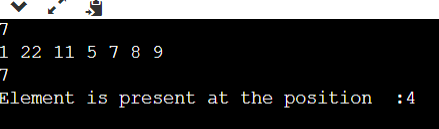
else

{

cout<<"Target element is not present in the array."<<endl;

}

}

INPUT AND OUTPUT: 

Question 7: Write a program to find a target element in a given array using Binary Search.

CODE:

#include<bits/stdc++.h> using namespace std; int binarySearch(int arr[],int key ,int start,int end)

{

if(end>=start)

{

int mid=start+(end-start)/2; if(arr[mid]==key)

{

return mid;

}

if(arr[mid]>key)

{

return binarySearch(arr,key,start,mid-1);

}

return binarySearch(arr,key,mid+1,end);

}

return -1;

}

int main()

{

int n;

cin>>n; int a[n];

for(int i=0;i<n;i++)

{

cin>>a[i];

}

int target; cin>>target; int status=binarySearch(a,target,0,n-1); if(status!=-1)

{

cout<<"Target elemet is present at the position :"<<status<<endl;

}

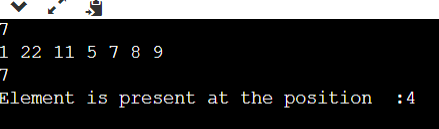
else

{

cout<<"Target element is not present in the array."<<endl;

}

}

INPUT AND OUTPUT: 

Question 8: Write a program to implement stack using array.

CODE:

#include<bits/stdc++.h> #define N 100 using namespace std;

class Stack

{

int top;

public: int arr[N];

Stack()

{

top=-1;

}

void Push(int val)

{

if(top==N)

{

cout<<"Stack OverFlow..!!!"<<endl; return;

}

arr[++top]=val;

}

int Pop()

{

if(top==-1)

{

cout<<"Stack Underflow..!!"<<endl; return -1;

}

return arr[top--];

}

bool isEmpty()

{

if(top==-1)

{

return true;

}

return false;

}

bool isFull()

{

if(top==N-1) return true;

return false;

}

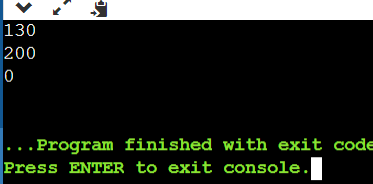
};

int main()

{

Stack st; st.Push(130); st.Push(200); st.Push(30); cout<<st.Pop()<<endl; cout<<st.Pop()<<endl; cout<<st.isEmpty()<<endl;

}

OUTPUT: 

**Question 9** : Write a program to implement a queue using an array.

CODE:

.#include<bits/stdc++.h> using namespace std; #define n 100 class Queue

{

public: int\* q; int front; int rear;

Queue()

{

q=new int[n]; front=-1; rear=-1;

}

bool isEmpty()

{

if(front==-1)

{

return true;

}

return false;

}

bool isFull()

{

if(rear==n-1)

{

return true;

}

return false;

}

void enQ(int val)

{

if(isFull())

{

cout<<"Queue is Full !!!"<<"\n"; return ;

}

if(isEmpty())

{

front++;

}

q[++rear]=val;

}

int deQ()

{

if(isEmpty())

{

cout<<"Queue is empty :"; return -1;

}

else

{

if(front!=rear)

{

return q[front++];

}

else

{

int temp=q[front]; front=-1; rear=-1; return temp;

}

}

}

};

int main()

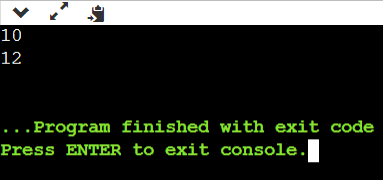
{

Queue Q;

Q.enQ(10); Q.enQ(12);

Q.enQ(30); cout<<Q.deQ()<<"\n"; cout<<Q.deQ()<<"\n";

}

OUTPUT: 

Question 10: Write a program to implement circular queue using array.

CODE:

#include<bits/stdc++.h> using namespace std;

class Queue

{

int rear,front; int size; int \*arr;

public:

Queue(int s)

{

front=rear=-1; size=s; arr=new int[s];

}

void enQueue(int value)

{

if ((front == 0 && rear == size-1) ||(rear == (front-1)%(size-1)))

{

cout<<"\nQueue is Full"<<endl;

return;

}

else if (front == -1)

{

front = rear = 0; arr[rear] = value;

}

else if (rear == size-1 && front != 0)

{

rear = 0; arr[rear] = value;

}

else

{

rear++; arr[rear] = value;

}

}

int deQueue()

{

if (front == -1)

{

printf("\nQueue is Empty"); return INT\_MIN;

}

int data = arr[front]; arr[front] = -1; if (front == rear)

{

front = -1; rear = -1;

}

else if (front == size-1)

front = 0;

else

front++;

return data;

} };

int main()

{

Queue q(5);

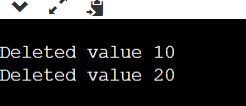
q.enQueue(10);

q.enQueue(20);

q.enQueue(30);

q.enQueue(40); cout<<"\nDeleted value "<< q.deQueue(); cout<<"\nDeleted value "<< q.deQueue();

}

OUTPUT: 

Question 11 Write a program to implement a stack using a linked list. CODE:

#include<bits/stdc++.h> using namespace std; class Node

{

public:

int data;

Node\* next;

Node(int val)

{

data=val; next=NULL;

}

};

class Stack

{

public:

Node\* top;

Stack()

{

top=NULL;

}

void Push(int val)

{

Node\* temp=new Node(val); if(top==NULL)

{

top=temp; return;

}

temp->next=top; top=temp;

}

void Pop()

{

if(top==NULL)

{

cout<<"Stack is empty..!"<<endl;

}

Node\* toDelete=top; top=top->next; toDelete->next=NULL; free(toDelete);

}

int Top()

{

if(isEmpty())

{

return -1;

}

return top->data;

}

bool isEmpty()

{

if(top==NULL)

{

return true;

}

return false;

}

};

int main()

{

Stack st; st.Push(10); st.Push(20); st.Push(30); st.Push(40); cout<<st.Top()<<endl; st.isEmpty()?cout<<"Stack is empty.!":cout<<"Not empty."; cout<<endl; st.Pop(); cout<<st.Top()<<endl;

}

OUTUT: 

Question 12: Write a program to implement a queue using a linked list.

Code:

#include<bits/stdc++.h> using namespace std; class node

{

public:

int data; node\* next;

node(int val)

{

data=val;

next=NULL;

}

};

class qu

{

node\* front; node\* back;

public: qu()

{

front=NULL; back=NULL;

}

void enQ(int val)

{

node\* n=new node(val); if(front==NULL)

{

back=n; front=n; return ;

}

back->next=n; back=n;

}

void deQ()

{

if(front==NULL)

{

cout<<"Queue Underflow..!!!"<<endl;

return;

}

node\* todelete=front; front=front->next; delete todelete;

}

int peek()

{

if(front==NULL)

{

cout<<"NO!!ELement in the Queue..!!"<<endl;

return -1;

}

return front->data;

}

bool isEmpty()

{

if(front==NULL)

{

return true;

}

return false;

}

};

int main()

{

qu Q; Q.enQ(1); Q.enQ(2);

Q.enQ(3); cout<<Q.peek()<<endl; Q.deQ();

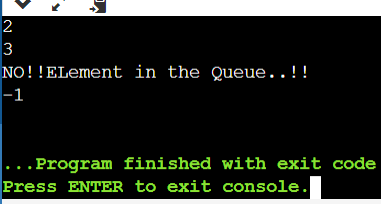
cout<<Q.peek()<<endl; Q.deQ();

cout<<Q.peek()<<endl; Q.deQ();

cout<<Q.peek()<<endl;

}

OUTPUT:



Question 13 : Write a program to implement Circular queue linked list.

Code:

#include<bits/stdc++.h>

using namespace std; class Node

{

public:

int data; Node\* next;

Node(int val)

{

data=val; next=NULL;

}

};

class cQueue

{

public:

Node\* front; Node\* rear;

cQueue()

{

front=NULL; rear=NULL;

}

void enQ(int val)

{

Node\* temp=new Node(val); if(front==NULL)

{

front=temp;

}

else

{

rear->next=temp;

}

rear=temp; rear->next=front;

}

int dQue()

{

if(front==NULL)

{

return -1; //Assuming that all nodes are non

negative.

}

else if(front==rear)

{

Node\* toDelete=front; int rt=toDelete->data; front=NULL; rear=NULL; toDelete->next=NULL; free(toDelete); return rt;

}

Node\* toDelete=front; int rt=toDelete->data; front=front->next; rear->next=front; toDelete->next=NULL; free(toDelete); return rt;

}

bool isEmpty()

{

if(front==NULL)

{

return true;

}

return false;

}

};

int main()

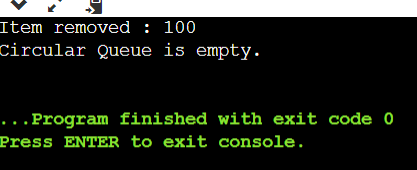
{

cQueue cq; cq.enQ(100); cout<<"Item removed : "<<cq.dQue()<<endl; cq.isEmpty()?cout<<"Circular Queue is empty."<<endl:cout<<"It is

not empty."<<endl;

}

OUTPUT:



Question 14: Write a program to implement a tree. Code:

#include<bits/stdc++.h> using namespace std; class Node

{

public: int data; Node\* left;

Node\* right;

Node(int val)

{

data=val; left=NULL; right=NULL;

}

};

int main()

{

Node\* root=new Node(1); root->left=new Node(2); root->right=new Node(3); root->left->left=new Node(4); root->left->right=new Node(5); root->right->left=new Node(6); root->right->right=new Node(7);

}

/\*

1

/\

2 3

/\ /\

4 5 6 7

\*/

Question 15: Write a program to traverse a binary tree.

Code:

#include<bits/stdc++.h> using namespace std; class Node

{

public: int data; Node\* left;

Node\* right;

Node(int val)

{

data=val; left=NULL; right=NULL;

}

};

void preorder(Node\* root)

{

if(root==NULL)

{

return ;

}

cout<<root->data<<" "; preorder(root->left); preorder(root->right);

}

void inorder(Node\* root)

{

if(root==NULL)

{

return ;

}

inorder(root->left); cout<<root->data<<" "; inorder(root->right);

}

void postorder(Node\* root)

{

if(root==NULL)

{

return;

}

postorder(root->left); postorder(root->right); cout<<root->data<<" ";

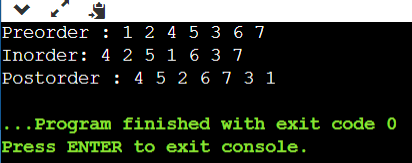
}

int main()

{

Node\* root=new Node(1); root->left=new Node(2); root->right=new Node(3); root->left->left=new Node(4); root->left->right=new Node(5); root->right->left=new Node(6); root->right->right=new Node(7); cout<<"Preorder : "; preorder(root); cout<<endl; cout<<"Inorder: "; inorder(root); cout<<endl; cout<<"Postorder : "; postorder(root);

}

OUTPUT: 

Question 16: Write a program to implement a Binary Search Tree (INSERTION AND DELETION).

Code:

#include<bits/stdc++.h> using namespace std;

class Node

{

public: int data; Node\* left;

Node\* right;

Node(int val)

{

data=val; left=NULL; right=NULL;

}

};

Node\* insertBst(Node\* root,int val)

{

if(root==NULL)

{

return new Node(val);

}

if(val<root->data)

{

root->left=insertBst(root->left,val);

}

else

{

root->right=insertBst(root->right,val);

}

return root;

}

void inorder(Node\* root)

{

if(root==NULL) return; inorder(root->left); cout<<root->data<< " "; inorder(root->right);

}

int main()

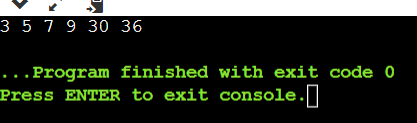
{

Node\* root=NULL; root=insertBst(root,5); insertBst(root,3); insertBst(root,30); insertBst(root,9);

insertBst(root,7); insertBst(root,36); inorder(root);

}

OUTPUT:



DELETION:

#include<bits/stdc++.h> using namespace std; class Node

{

public: int data; Node\* left;

Node\* right;

Node(int val)

{

data=val; left=NULL; right=NULL;

}

};

Node\* inorderSuccessor(Node\* root)

{

Node\* curr=root; while(curr&&curr->left!=NULL)

{

curr=curr->left;

}

return curr;

}

Node\* deleteInBst(Node\* root,int key)

{

if(key<root->data)

{

root->left=deleteInBst(root->left,key);

}

else if(key>root->data)

{

root->right=deleteInBst(root->right,key);

}

else

{

if(root->left==NULL)

{

Node\* temp=root->right; free(root); return temp;

}

else if(root->right==NULL)

{

Node\* temp=root->left; free(root); return temp;

}

else{

Node\* temp=inorderSuccessor(root->right); root->data=temp->data; root->right=deleteInBst(root->right,temp->data);

}

}

return root;

}

void inorder(Node\* root)

{

if(root==NULL)

{

return;

}

inorder(root->left); cout<<root->data<<" "; inorder(root->right);

}

int main()

{

Node\* root=new Node(4); root->left=new Node(2); root->right=new Node(5); root->left->left=new Node(1); root->left->right=new Node(3); root->right->right=new Node(6);

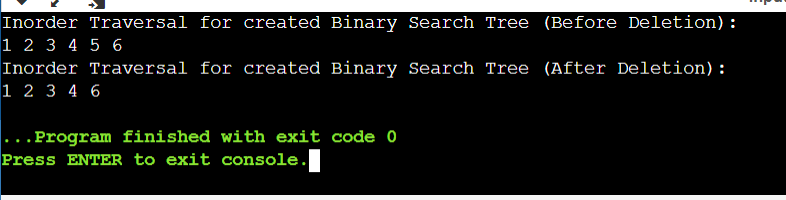
cout<<"Inorder Traversal for created Binary Search Tree (Before

Deletion): "<<endl; inorder(root); cout<<endl; root=deleteInBst(root,5);

cout<<"Inorder Traversal for created Binary Search Tree (After

Deletion): "<<endl; inorder(root);

}

OUTPUT: 

Question 15: Write a program to implement BFS and DFS traversal technique.

Code:

#include<bits/stdc++.h> using namespace std; void BFS(int s,vector<int>adj[],vector<int>&vis)

{

queue<int>q; q.push(s); vis[s]=1; while(!q.empty())

{

int curr=q.front(); q.pop(); cout<<curr<<" "; for(auto it:adj[curr])

{

if(!vis[it])

{

q.push(it); vis[it]=1;

}

}

}

}

int main()

{

int n,m;

cin>>n>>m; vector<int> adj[n+1]; for(int i=0;i<m;i++)

{

int u,v;

cin>>u>>v; adj[u].push\_back(v); adj[v].push\_back(u);

}

vector<int>vis(n+1,0); cout<<"Traversing by using BFS : "<<endl; for(int i=1;i<=n;i++)

{

if(!vis[i])

{

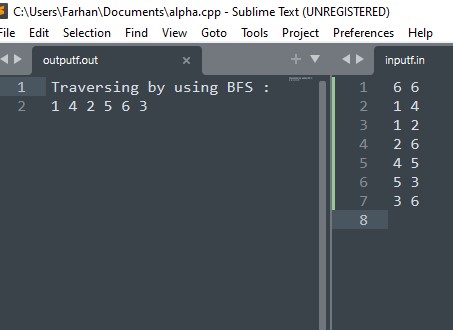
BFS(i,adj,vis);

}

}

}

INPUT AND OUTPUT:



DFS Traversal: #include<bits/stdc++.h> using namespace std;

void DFS(int s,vector<int>adj[],vector<int>&vis)

{

cout<<s<<" "; vis[s]=1; for(auto it:adj[s])

{

if(!vis[it])

{

DFS(it,adj,vis);

}

}

}

int main()

{

int n,m;

cin>>n>>m; vector<int>adj[n+1]; for(int i=0;i<m;i++)

{

int u,v;

cin>>u>>v; adj[u].push\_back(v); adj[v].push\_back(u);

}

vector<int>vis(n+1,0); cout<<"Traversal by using DFS : "<<endl; for(int i=1;i<=n;i++)

{

if(!vis[i])

{

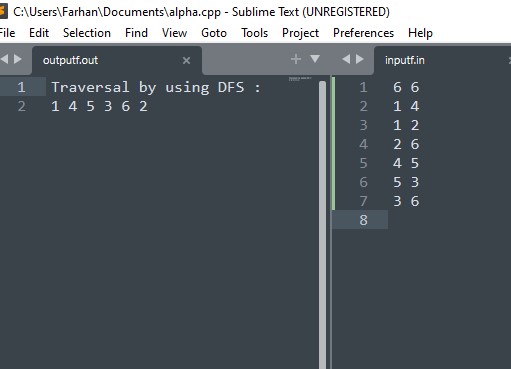
DFS(i,adj,vis);

}

}

}

INPUT AND OUTPUT:



Question 16: Write a program to Construct a minimum cost spanning tree.

Code:

#include<bits/stdc++.h> using namespace std;

// KRUSKAL ALGORITHM USED. class Node

{

public: int u,v,w;

Node(int x,int y,int vv)

{

u=x; v=y; w=vv;

}

};

bool comp(Node a,Node b)

{

return a.w<b.w;

}

int findParent(int n,vector<int>&parent)

{

if(parent[n]==n)

{

return n;

}

return parent[n]=findParent(parent[n],parent);

}

void unionn(int u,int v,vector<int>&parent,vector<int>&rank)

{

int p=findParent(u,parent); int q=findParent(v,parent); if(rank[p] < rank[q]) {

parent[p] = q;

}

else if(rank[q] < rank[p]) {

parent[q] = p;

}

else {

parent[q] = p; rank[p]++;

}

}

int main()

{

int n,m;

cin>>n>>m; vector<Node> edges; for(int i=0;i<m;i++)

{

int u,v,w;

cin>>u>>v>>w; edges.push\_back(Node(u,v,w));

}

sort(edges.begin(),edges.end(),comp); vector<int>parent(n+1); for(int i = 0;i<=n;i++)

parent[i] = i;

vector<int> rank(n+1, 0);

int cost = 0; vector<pair<int,int>> mst; for(auto it : edges) { if(findParent(it.v, parent) != findParent(it.u, parent)) {

cost += it.w; mst.push\_back({it.u, it.v}); unionn(it.u, it.v, parent, rank);

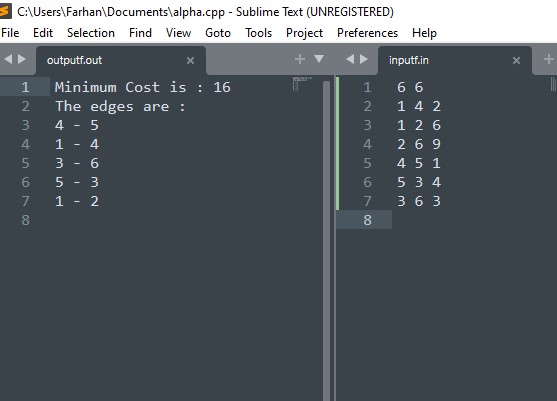
}

}

cout <<"Minimum Cost is : "<< cost << endl; cout<<"The edges are : "<<endl; for(auto it : mst) cout << it.first << " - " << it.second << endl;

}

INPUT AND OUTPUT:



Question 17 : Write a program to illustrate the shortest path in graph.

Code:

#include<bits/stdc++.h> using namespace std;

// DIJKSTRA ALGORITHM

int main(){

int n,m,source; cin >> n >> m; vector<pair<int,int> > g[n+1];

int a,b,wt; for(int i = 0; i<m ; i++){ cin >> a >> b >> wt; g[a].push\_back(make\_pair(b,wt)); g[b].push\_back(make\_pair(a,wt));

}

cin >> source; priority\_queue<pair<int,int>,vector<pair<int,int> >,greater<pair<int,int>

> > pq; vector<int> distTo(n+1,INT\_MAX);

distTo[source] = 0; pq.push(make\_pair(0,source));

while( !pq.empty() ){ int dist = pq.top().first; int prev = pq.top().second; pq.pop();

vector<pair<int,int> >::iterator it; for( it = g[prev].begin() ; it != g[prev].end() ; it++){ int next = it->first; int nextDist = it->second; if( distTo[next] > distTo[prev] + nextDist){ distTo[next] = distTo[prev] + nextDist; pq.push(make\_pair(distTo[next], next));

}

}

}

cout << "The distances from source, " << source << ", are : \n"; for(int i = 1 ; i<=n ; i++) cout << distTo[i] << " "; cout << "\n";

return 0;

}

INPUT AND OUTPUT:

